In the Claims:

Please amend claim 78 as follows. All other claims remain unchanged. Please add new claims 90-93.

27. (Previously presented) The method of claim 71 wherein the client library determines the server application based on an association with the class identifier.

28. (Previously presented) The method of claim 27 wherein the association is recorded during installation of the server application.

29. (Previously presented) The method of claim 27 wherein the association is recorded when the server application is launched.

30. (Previously presented) The method of ~~claim~~ 27 including when the server application supports an object that is compatible with the client application, launching the server application.

31. (Previously presented) The method of claim 30 wherein the client application is executing in a process and the server application is launched in a separate process.

32. (Previously presented) The method of claim 30 wherein the client application is executing in a process and the server application is launched in the same process.

33. (Previously presented) The method of claim 30 wherein the client application and the server application exchange data using a compatible format.

34. (Previously presented) The method of claim 27 wherein the client library determines the association while the server application is not executing.

35. (Previously presented) The method of claim 71 wherein the server application records in the configuration store, an association between itself and the class identifier.

36. (Previously presented) The method of claim 35 including when the server application supports a data format that is compatible with the client application, launching the server application.

37. (Previously presented) The method of claim 36 wherein the client application is executing in a process and the server application is launched in a separate process.

38. (Previously presented) The method of claim 36 wherein the client application is executing in a process and the server application is launched in the same process.

39. (Previously presented) The method of claim 36 wherein the client application and the server application exchange data using a compatible format.

40. (Previously presented) The method of claim 71 wherein the client application, the client library, the server application, and the server library, each execute in a separate process.

41. (Previously presented) The method of claim 71 for supplying the server application to perform the requested manipulation wherein the server application populates the configuration store with class identifiers it supports.

42. (Previously presented) The method of claim 41 wherein the client application determines the server application that supports the class identifier while the server application is not executing.

43. (Previously presented) The method of claim 41 wherein the server application populates the configuration store during installation of the server application.

44. (Previously presented) The method of claim 41 wherein the server application populates the configuration store when the server application is launched.

45. (Previously presented) The method of claim 41 including when the server application supports a data format that is compatible with the client application, launching the server application.

46. (Previously presented) The method of claim 45 wherein the client application is executing in a process and the server application is launched in a separate process.

47. (Previously presented) The method of claim 45 wherein the client application is executing in a process and the server application is launched in the same process.

48. (Previously presented) The method of claim 45 wherein the client application and the server application exchange data using a compatible format.

49. (Previously presented) A computer-readable medium containing instructions for causing a computer system to perform the method of claim 71.

71. (Previously presented) A method in a computer system for manipulating an object displayable in a client application via any one of a plurality of server applications using an application programming interface, the computer system having a configuration store for storing a class identifier associated with the object and associating the class identifier with at least one of the server applications of the plurality of server applications, the method comprising:

requesting by the client application through the application programming interface, a manipulation to be performed on the object wherein routines of the application programming interface are divided into an object-independent client library and a server library, the object-independent client library comprising routines which invoke the proper server application to manipulate the object, and the server library comprising routines which process requests to manipulate the object;

determining by the object-independent client library using the configuration store and the class identifier of the object, a server application out of a plurality of server applications to perform the requested manipulation on the object;

sending by the object-independent client library, a message to the server library to perform the requested manipulation on the object;

receiving by the server library the message to perform the requested manipulation on the object; and

invoking by the server library the server application to perform the requested manipulation on the object.

72. (Previously presented) The method of claim 71, wherein the object displayable in the client application is a first object, the method further comprising:

depicting the first object as appearing inside a second object displayable in the client application.

73. (Previously presented) The method of claim 71, wherein the client application determines from the configuration store and displays for a user a list of available manipulations on the object.

74. (Previously presented) The method of claim 71, wherein the server application is started up in response to receiving the message.

75. (Previously presented) The method of claim 71, wherein the server application shuts down after completion of manipulations requested in the message.

76. (Previously presented) The method of claim 71 wherein a user can select a new object from amongst a plurality of embedded or linked objects displayed in a graphical user interface.

77. (Previously presented) The method of claim 71 wherein a user can select a manipulation or procedure to be performed on a selected object from amongst a plurality of manipulations or procedures displayed in a graphical user interface.

78. (Currently amended) A method **running on** [in] a **single** computer [system], the computer [system] having a configuration store for storing identifiers of available embedded or linked objects and identifiers of servers associated with the embedded or linked objects, the method comprising:

providing an application programming interface supporting embedded or linked objects wherein the application programming interface is separated into an object-independent client library and a server library, the object-independent client library comprising routines which determine servers to manipulate embedded or linked objects, and the server library comprising routines which invoke server routines on embedded or linked objects;

requesting by a user from a client application, creation of an embedded or linked object;

requesting by the client application via the application programming interface to the object-independent client library, available embedded or linked objects;

determining by the object-independent client library from the configuration store and returning to the client application, a list of the available linked or embedded objects;

presenting by the client application to the user, the available embedded or linked objects;

selecting by the user from the available presented objects an object to be linked or embedded within a container object displayed via the client application;

requesting by the client application via the application programming interface to the object-independent client library, creation of the selected embedded or linked object;

determining by the object-independent client library from the configuration store a server associated with the selected linked or embedded object and sending a message from the object-independent client library to the server library, to create the selected linked or embedded object;

receiving by the server library, the message to create the selected embedded or linked object; and

invoking by the server library at the determined server, a server routine to create the selected embedded or linked object;

whereby the created linked or embedded objected is created by the server routine and the user can edit or otherwise manipulate the created linked or embedded object within the container object.

79. (Previously presented) The method of claim 78 wherein the user is able to edit or manipulate a linked or embedded object by selecting an action available on a client application menu.

81. (Previously presented) A computer software system comprising:

client means for displaying containee objects within the client means, wherein containee objects are associated with object class identifiers;

plural server means for performing manipulations on containee objects;

configuration store means for storing an association between object class identifiers and server means that perform manipulations on associated object classes;

client library means for receiving from the client means, requests to perform manipulations on containee objects, consulting the configuration store means and determining an appropriate server means out of the plural server means to perform manipulations based on object class identifiers, and sending messages to server library means associated with determined server means, said messages comprising requests to perform manipulations on containee objects; and

server library means for receiving messages to perform requested manipulations, and invoking server routines for performing requested manipulations.

82. (Previously presented) A computer-readable medium containing instructions for causing a computer system to perform the method of claim 27.

83. (Previously presented) A computer-readable medium containing instructions for causing a computer to perform the method of claim 78.

84. (Previously presented) The method of claim 71 further comprising:

receiving by the server library, an indication from the server application that the requested manipulation is complete;

sending a message from the server library that the requested manipulation is complete;

receiving by the client library, the message from the server library that the requested manipulation is complete; and

sending an indication to the client application that the requested manipulation is complete.

85. (Previously presented) The method of claim 71 wherein the client application and the client library are dynamically linked to execute in the same process.

86. (Previously presented) The method of claim 71 wherein the server application and the server library are dynamically linked to execute in the same process.

87. (Previously presented) The method of claim 71 wherein the client library and the server library send messages via a channel comprising inter-process communication.

88. (Previously presented) The method of claim 71 wherein the client application, the client library, the server application, and the server library are processes sharing the same processor.

89. (Previously presented) The method of claim 71 wherein the application programming interface provides functions comprising compound document functionality.

90. (New) A single computer comprising plural processes comprising:

a client process; and

plural server processes;

wherein the client process comprises a client application and a client library;

the plural server processes each comprises a server application and a server library; and

the client library sets up message passing connections to each server process via inter-process communications.

91. (New) The single computer of claim 90, wherein the client library and server library provide an interface through which the details of inter-process communications are shielded from the client and server applications.

92. (New) The single computer of claim 90, wherein each server process implements a particular type of containee object that is opened by the client process.

93. (New) A method in a computer system for manipulating an object displayable in a client application via any one of a plurality of server applications using an application programming interface, the computer system having a configuration store for storing a class identifier associated with the object and associating the class identifier with at least one of the server applications of the plurality of server applications, the method comprising:

requesting by the client application through the application programming interface, a manipulation to be performed on the object wherein routines of the application programming interface are divided into an object-independent client library and a server library, the object-independent client library comprising routines which invoke the proper server application to manipulate the object, and the server library comprising routines which process requests to manipulate the object;

determining by the object-independent client library using the configuration store and the class identifier of the object, a server application out of a plurality of server applications to perform the requested manipulation on the object;

sending by the object-independent client library, a message to the server library to perform the requested manipulation on the object;

receiving by the server library the message to perform the requested manipulation on the object;

invoking by the server library the server application to perform the requested manipulation on the object;

upon completing the requested manipulation, the server application returns to the server library, and the server library sends a message to the client library indicating that the manipulation is complete; and

the client library receives the message and indicates to the client application that the manipulation is complete.